If only I can trust my police! SIM : an agent-based audit solution of access right deployment through open network

Incoul, Christophe; Gateau, Benjamin; Aubert, Jocelyn; Bounoughaz, Nicolas; Feltus, Christophe

Published in:
Proceedings of International Conference on Risks and Security of Internet and Systems (CRiSIS 2008), Tozeur, Tunisia

DOI:
10.1109/CRISIS.2008.4757467

Publication date:
2008

Document Version
Early version, also known as pre-print

If only I can trust my police!
SIM : an agent-based audit solution of access right deployment through open network

Christophe Incoul, Benjamin Gateau, Jocelyn Aubert, Nicolas Bounoughaz, Christophe Feltus
Centre for IT Innovation
Centre de Recherche Public Henri Tudor
29, Rue John F. Kennedy
L-1855 Luxembourg
christophe.incoul@tudor.lu

Abstract

Dynamic and evolved environment make the Information Systems (IS), and consequently access rights to its components, always more complex to define and to manage. This statement is mainly explained by the continuous grow of the diversity of business requirements and by the criticality of the resources to protect. Even if a proliferation of sophisticated "Identity and Access Management" (IAM) solutions has appeared on the market since end of last decade, some points remain poorly addressed like the definition of the access control policy against business constraints and their dissemination through distributed system.

To bring up a contribution for improving that matter, our paper's first objective is to realize the development of an automate deployment of policies from an administrative platform that encompasses business requirements down to infrastructure's components and devices. This objective is achieved by adapting the XACML OASIS framework [22] and by formalizing a protocol for information exchange through different components of a multi-agent system.

The second paper's objective aims at providing guaranties that defined and deployed access rights are continuously aligned with business requirements. This objective is completed by complementary developments that aim to perform a systematic and/or on-demand audit of the effective rights against the desired ones. This second objective is achieved by adding new functionality to the proposed agents architecture and by adapting the protocol accordingly.

Practically, this research has been performed in the framework of the SIM [1] project and has privileged free and open source components for the prototyping phase.

Keywords: Identity Management, Responsibility model, Policy audit, multi agent architecture.

1. Introduction

Improving access rights deployment and giving business manager the confidence that rights are correctly enforced is our research's aim. That twofold objective is nowadays challenging because the configuration of Information System has been subject to major changes since the apparition of open and distributed network. What was previously a rather simple manageable administrative task is now a work that takes considerable proportions. This assertion is mainly due to two following statements. Firstly, the management of access right over business assets was previously the responsibility of the IT staff and is now hand over the responsibility of business owners. This shift of responsibility seems reasonable in that it is the business that has to define which stakeholders need to access which resources. However, because business manager are not friendly with so call "unintelligible" IT applications, it is necessary to provide them adapted and clear user interfaces. First results of SIM project have focused on the elaboration of such interfaces by using an open source framework named eGroupWare [5]. Secondly, the management of access right that was previously limited to a strict company environment has evolved toward a wild opening. Resources to be accessed are no more only located on a closed network but may be posted on servers based on the other side of the world. Likely, people that need to access corporate information are no more limited to employees of the company but is largely open to others stakeholders like for instance shareholders that need financial information, providers that check the state of stock or customers that follow on-line the state of orders.

Based upon that observation, it appears that it is unavoidable to have a trusted access control framework without previously having defined clear responsibility for each stakeholder, provisioning access rights accordingly to all IS components and devices, and finally auditing that those rights are suitably applied.

Defining such a framework remains however challenging because of the difficulty to integrate heterogeneous applications - consequently technologies - to heterogeneous organizations.

As shown on Figure 1 identity management is an activity that could be achieved following a life cycle approach. First results of our research attempt to bring innovation to parts "Policy Engineering", "Policy Deployment" and "Policy audit".

The section 2 of this paper proposes a responsibility model designed to be comprehensible by business manager while offering at the same time pragmatic information to IT staff. To keep the paper didactic, a case study is introduced at the early beginning of the section to illustrate the concepts of the model. The Section 3 presents the business interface for responsibilities and access rights management. Section 4 presents the agents based solution for the deployment of rights through the network and the audit of those rights. Finally, section 5 introduces future work and concludes.

Figure 1: Identity management life cycle

2. Responsibility model

Our previous works [1] have presented responsibility model (cf. Figure 2) and more precisely how it has been elaborated according to a literature review and by confrontation to others theories.

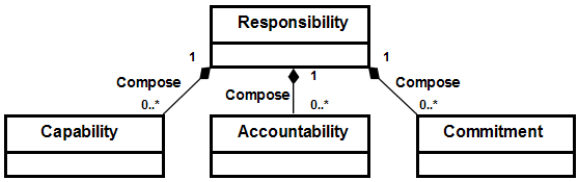

Figure 2: Responsibility model

To introduce this model, we proprietary propose the following case study and explain concepts by providing illustrations related to it.

Mister Johnson is the manager of the IT Company named "HighTech". Each year, Mister Johnson organizes during the Christmas period a large sending of postcards to all its customers. This year, Mister Johnson has too much work for closing the annual report and consequently decides to delegate this task to one of its employees. Because the task is less business sensitive as some other production task, Mister Johnson decides to delegate it to a part-time secretary named Miss Fleming. Miss Fleming has just got married and consequently, she accepts this additional work without commitment. Mister Johnson asks to the IT service manager to give Miss Fleming the necessary access right to the customers address list. The IT service manager asks an employee from the IT service named Rob to realize the necessary operation for providing this right. On January the 30th, Mister Johnson receives over 100 complains of customers that didn't receive Christmas card.

Mister Johnson has duly formalized Miss Fleming's Accountability by asking her to realize the sending activity. It was consequently clear about what she was accountable to do. To achieve that sending, she got the necessary capability that was the access to the customers file. However, due to the fact that her thought went to her new husband rather that to the work she had to accomplish, she didn't really want to achieve the work and failed to assure her responsibility due to a miss of commitment.

Rob's responsibility can also be analyzed by that case study. Rob is a well paid IT staff that is very happy with his function. He has received clear accountability to give access right to Miss Fleming and he has the needed capabilities due to its position as network administrator. He has consequently been responsible to fulfil Mister Johnson' request.

It exists a plethora of definitions of responsibility and this paper has not for duty to propose a new one. We may however state that commonly accepted responsible definition encompasses the idea of having the obligation to ensure that something happens. Moreover, the review of the literature in [2] shows that it makes sense to hang on to it the three additional elements that are Capability, Accountability and Commitment. The relationship between Responsibility and Capacity, Accountability and Commitment is of the form 0..* to 1. That means that being responsible involves that it is possible to dispose of many Capacities, Accountabilities and Commitment. But at the opposite, on Commitments is only bound to one responsibility, and adequately for Accountability and Capability.

Capability describes the quality of having the requisite qualities, skills or resources to perform a task. Capability is a component that is part of all models and methods, and is most frequently declined through definition of access rights, authorizations or permissions. Based upon the above case study, the Capability is illustrate through the Miss Fleming's capability to access the customer's file. This Capability exists because Rob was responsible to provide that access right. The case study illustrates also Rob's Capability to be responsible for providing access right. Indeed, due to his position of network administrator, he has the right to manage all employees' access right.

Accountability is a concept that exists mainly in requirement engineering methods and that appears through the obligation to achieve a task or to perform an action. This concept describes the state of being answerable about the achievement of a task. The above case study illustrates that Miss Fleming is accountable toward Mister Johnson regarding the task she has been assigned responsible for. In the same way, Rob is accountable toward the IT service manager for providing the access right.

Commitment is the moral engagement of a stakeholder to fulfil a task and the assurance that he will

do it. Commitment is a most infrequent concept. Traditional policy model such as RBAC [3] do not address it, however i* [4] partly introduces it (e.g. when defining dependency as an "agreement" between two actors). However, to distinguish if it is a moral concept or an obligation remains interpretable. This component is illustrated through the cases study as follow: Firstly, we may state that because Miss Fleming has others duties in mind, she has not the willingness to achieve the task. We may state that she is not committed to do it. At the opposite, Rob is a well paid IT staff that is very happy with his function. He is fully committed to perform the task.

3. Business interface for responsibilities and access rights management

In order to support our approach, we have developed a prototype, using the open-source groupware eGroupWare, which allows defining business' processes on which responsibilities are assigned to stakeholders.

3.1 Responsibility enforcement

Using this paper's case study, the first step is to define the process "*XMAS-MAILING-2007 - Christmas card mailing – Year 2007" (*cf. Figure 3).

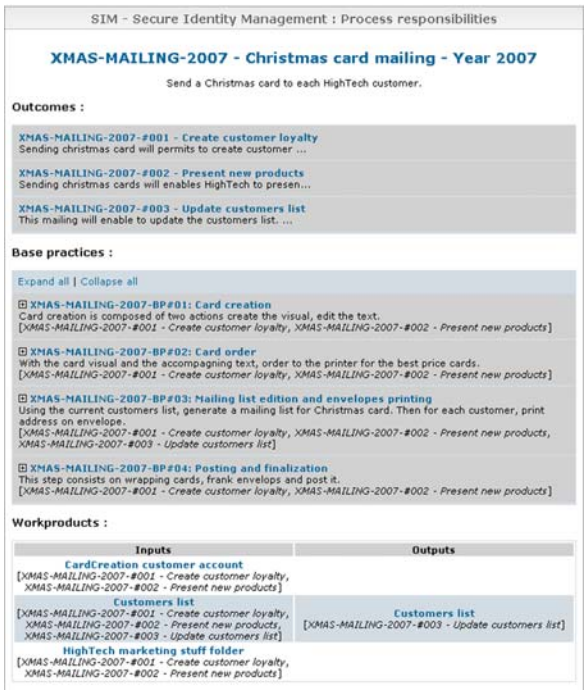

Figure 3: SIM prototype process cartography

The process defined different outcomes, which can be defined as results produced by the process :

Outcome #01 : *Create customer loyalty*
Outcome #02 : *Present new products*
Outcome #03 : *Update customers list*

Outcomes are reached by achieving base practices (BP) :

XMAS-MAILING-2007-BP#01 : Card creation
XMAS-MAILING-2007-BP#02 : Card order
XMAS-MAILING-2007-BP#03 : Mailing list edition and envelopes printing
XMAS-MAILING-2007-BP#04 : *Posting and finalization*

Outcomes are reached by using some work products (WP) :

WorkProduct#1 : CardCreation customer account
WorkProduct#2 : Customers list
WorkProduct#3 : HighTech marketing stuff folder

For a better understanding and granularity, we defined a base practice as a set of atomic actions, called actions, and we define responsibilities for those actions.

For our case study, we define for example a responsibility on the action "Edit mailing list" which is a part of the base practice "XMAS-MAILING-2007-BP#03: Mailing list edition and envelopes printing". This responsibility is assigned to Miss Fleming and is composed of two accountabilities "Create a relevant customers list for card mailing based on customers list" and "Modify obsolete entries in customers list", and one capability (to edit the customer list, she needs to "Access customers list on read-write mode"). Each responsibility is created using the form showed on Figure 4.

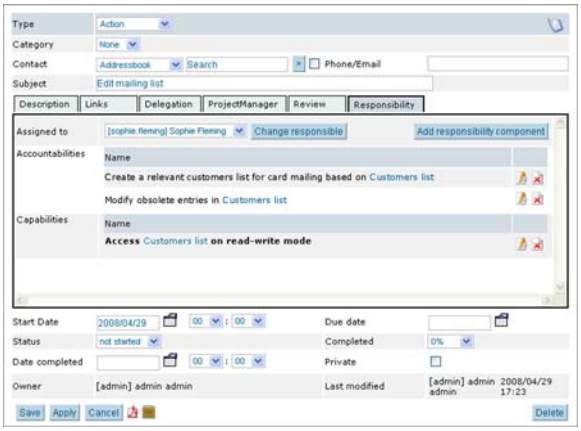

Figure 4: SIM action's responsibilities add form

When all responsibilities are defined and assigned to resources, the application, using these responsibilities, is able to publish via a web-service, a set of XACML policies containing all process related policies (Figure 5 presents the policy set corresponding to defined responsibilities). These technical mechanisms of rights enforcement are detailed in section 5.

3.2 Audit module

Once the deployment of the access rights is done on the technical devices via the multi-agent system, we need a mean to control, at the organizational layer, that polices are effectively and rightly deployed and applied at the technical layer to:

- *Ensure a high level of effectiveness in the policy deployment process;*
- *Ensure a high level of correlation between the business policies issued from the organizational model down to accesses rights enforced at the technical devices;*

```
<?xml version="1.0" encoding="UTF-8"?>
<PolicySet xmlns="urn:oasis:names:tc:xacml:1.0:policy"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="urn:oasis:names:tc:xacml:1.0:policy cs-xacml-schema-policy-01.xsd"
  PolicySetId="policySetForProcess15"
  PolicyCombiningAlgId="urn:oasis:names:tc:xacml:1.0:policy-combining-algorithm:first-applicable">
  <Target>
    <Subjects><AnySubject/></Subjects>
    <Resources><AnyResource/></Resources>
    <Actions><AnyAction/></Actions>
  </Target>
  <Policy xmlns="urn:oasis:names:tc:xacml:1.0:policy"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:oasis:names:tc:xacml:1.0:policy cs-xacml-schema-policy-01.xsd"
    PolicyId="Policy_for_subject_26_on_resource_31"
    RuleCombiningAlgId="urn:oasis:names:tc:xacml:1.0:rule-combining-algorithm:deny-overrides">
    <Description>Allow subject 26 to read resource 31</Description>
    <Target>
      <Subjects><AnySubject/></Subjects>
      <Resources><AnyResource/></Resources>
      <Actions><AnyAction/></Actions>
    </Target>
    <Rule RuleId="Rule_for_subject_26_on_resource_31" Effect="Permit">
      <Description>Allow subject 26 to read resource 31</Description>
      <Target>
        <Subjects>
          <Subject>
            <SubjectMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:integer-equal">
              <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#integer">26</AttributeValue>
              <SubjectAttributeDesignator
                AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"
                DataType="http://www.w3.org/2001/XMLSchema#integer"/>
            </SubjectMatch>
          </Subject>
        </Subjects>
        <Resources>
          <Resource>
            <ResourceMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:integer-equal">
              <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#integer">31</AttributeValue>
              <ResourceAttributeDesignator
                AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id"
                DataType="http://www.w3.org/2001/XMLSchema#integer"/>
            </ResourceMatch>
          </Resource>
        </Resources>
        <Actions>
          <Action>
            <ActionMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
              <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">read</AttributeValue>
              <ActionAttributeDesignator
                AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"
                DataType="http://www.w3.org/2001/XMLSchema#string"/>
            </ActionMatch>
          </Action>
          <Action>
            <ActionMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
              <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">write</AttributeValue>
              <ActionAttributeDesignator
                AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"
                DataType="http://www.w3.org/2001/XMLSchema#string"/>
            </ActionMatch>
          </Action>
        </Actions>
      </Target>
    </Rule>
  </Policy>
</PolicySet>
```

Figure 5: XACML Policy set generated by SIM prototype

To reach these goals, we have developed an audit module that enables IT administrators and business managers to continually check the alignment of the access right with business' requirements. This monitoring is facilitated by the use of dashboards that highlight the policy deployment status through the mean of charts and diagrams. With these charts, administrators can detect problems induced by a modification of (technical or business) access rights and thus mitigate the risk of possible impacts on the security of the Information System.

The audit mechanism is illustrated through our case study by Figure 6 that gives a detailed view of the result of the audit of the *"XMAS-MAILING-2007 - Christmas card mailing – Year 2007"* process deployment. We can observe that the deployment of the access right defined for the action *"Print of B&WPrinterXYZ-CD2014"* is not correctly deployed on the specific device and we can see the reason why by hitting the "*error details*" link.

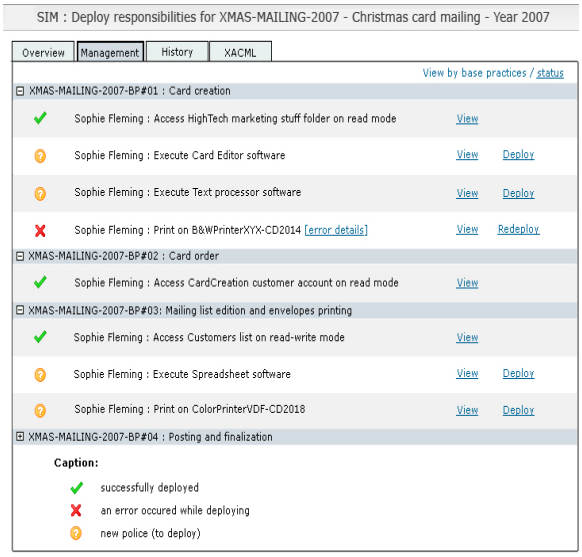

Figure 6: Example of deployment result for the case study

Each action has an indicator that represents the "access right status" for the action. We have defined three possible states:

1. *"Successfully deployed"*, if the access right is successfully implemented on the technical device;
2. *"An error occurred while deploying"*, if a problem has been encountered during the deployment process;
3. *"New police"*, if the police has never been deployed yet, or has changed (on the technical device or in the business layer) since last deployment.

For all actions, we can visualized the XACML policy linked to the right defined. For each action "*in error*" or "*not yet deployed*", we can deploy the access right policy individually. For each action "*in error*", the error message is available.

The second view, presented in Figure 7, gives a consolidated view of the state of the policies defined for

our process. Unsophisticated formulas have been used to generate the graphics but they are not detailed in that paper because it is not valuable at this stage of the research.

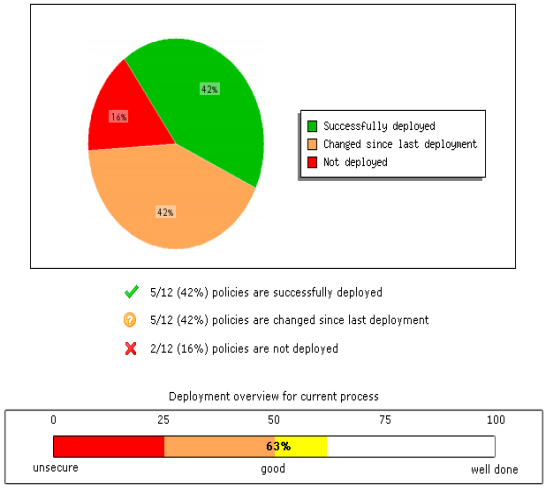

Figure 7: Consolidated view for the process

These two dashboards are obtained by comparing information retrieved from the deployment process and from the business requirements definition.

The next section explains in details the architecture of the policy deployment and audit process.

4. Policy Deployment and Audit

We need a means to transform an instantiated policy (composed of concrete rules) into specific commands to apply on concerned devices (named hereafter *technical modules*), to verify that the policy is applied with success and to check that no modification is directly done through the technical modules. We distinguish two phases.

The first one is the deployment:

1. We must find all the devices (*firewall* in our case study) concerned by the policy's rules.
2. The rules must be sent to the technical modules.
3. Each received rules must be transformed into script or command.
4. Scripts or commands must be executed and return an execution status.
5. An audit is done and sent back to the organisational layer in order to verify that policies have really been applied.

The second phase is the audit:

1. The access rights defined for a user or a resource must be checked.
2. The request is sent to the technical modules that transform it into command.
3. Technical modules execute the command and result is sent back to the user.

For that, several components are used (cf. Figure 8). Each technical module is interfaced with a Policy Enforcement Point (PEP). The PEP communicates with a component called Policy Decision Point (PDP) whose goal is to retrieve PEP and distributing rules to be applied. It also interfaces the policy base in order to be aware of new policies to apply. The PEP also communicates with a component called Audit Correlation Engine (ACE) whose goal is to get the status of PEP in general and the status of policies deployed in particular.

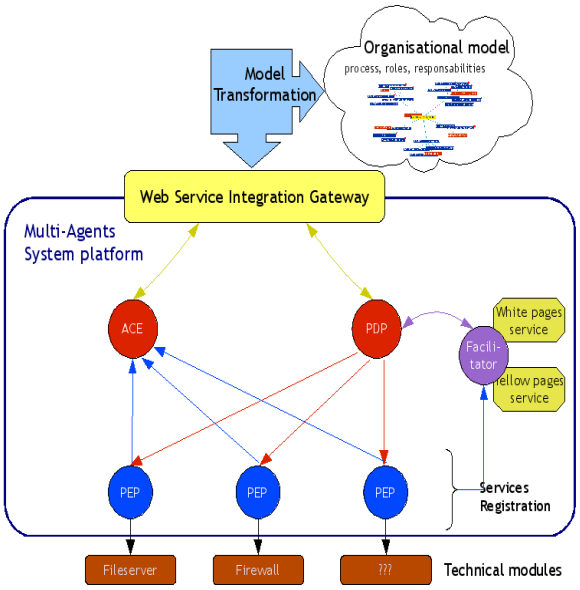

Figure 8: Technical infrastructure

The communication between the components could be provided by a standardized protocol such as SNMP [11], COPS [9] or NETCONF [10] or a multi-agent based communication.. We presented these different solutions and argued in favour or multi-agent system in [1]. Our conclusion was that we think that the use of a Multi-Agent System (MAS) is an interesting solution because it provides autonomous entities that can be collaborative. A Multi-Agent System is composed of several agents, capable of a mutual interaction that can be in the form of message passing or the production of changes in their common environment [6]. Agents are pro-active, reactive and socially autonomous entities able to exhibit organized activity, in order to meet their designed objectives, by eventually interacting with users. Agents are collaborative by being able to commit themselves to the society or/and another agent [7]. So, if we consider that each technical module is interfaced with an agent, all agents will collaborate in order to apply a set of common policies.

We detail in the following agents' architecture representing all components (PDP, PEP and ACE) and the relation between these components.

4.1. Policy Decision Point

The PDP's architecture is shown in Figure 9. There are two main modules: the *policy analysis* and the *Component Configuration Mapper*.

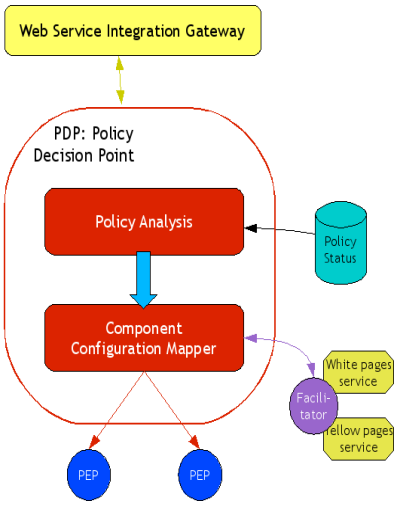

Figure 9: Policy Decision Point architecture

The policy analysis module has to perform a variety of validation checks. First, it verifies the syntax of the policy specification provided by a PIE. This module will then verify that the newly received policies are *consistent* with current applied rules (coming from the policy status base). A set of policies will be consistent if it can be shown that no contradictory policies will ever be found in a SIM system. The user will be able to choose the system behaviour if a conflict is detected. For the moment, the old rules that derivate from the previous policy are cancelled and the newly received policy that contradicts the applied rules.

The policy analysis module communicates with a "policy rules status" database. This database stores the newly received policies and their current status (in progress, not applicable, by-passed, enforced, removed…). In addition, the module should detect rules that cannot be enforced due to a lack of PEP. As a consequence a PDP should be aware of the different managed PEPs.

For this reason, a Facilitator agent helps the PDP agent. This agent manages the network topology by retrieving PEP agents according to their localisation (devices registered with an IP address or MAC address) or according to actions they could apply and their type (firewall, fileserver, etc.). For this, the *Facilitator* uses white pages and yellow pages services.

The Component Configuration Mapper states in details which kind of actions need to be taken by which kind of network devices/applications. This module receives high level policies and generates generic format policies for each type of PEP (router, firewall, IDS…). For that, it asks the Facilitator to determine what PEPs are impacted by the policies update by mapping a set of possible actions to the current network components capabilities.

If some rules are not applicable, the Component Configuration Mapper notifies the policy analysis module. This one will update the policy rules status. Problematic rules will be passed by, and their status in the "policy status" database will change from "in progress" to "by-passed". Then the corresponding policies are sent to the concerned PEP.

4.2. Policy Enforcement Point

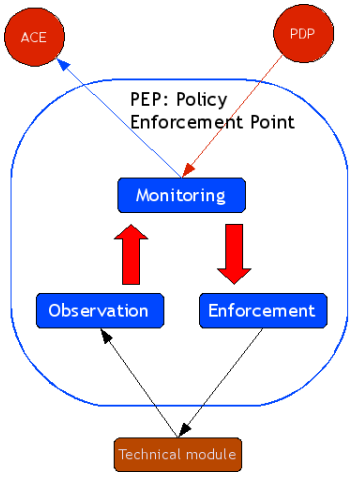

Figure 10: Policy Enforcement Point architecture

A PEP agent manages each device that is part of SIM's technical layer. Agents are specific according to the kind of devices or the kind of services that the device offers. It is specific in order to know how to transform policies represented in an abstract format (XACML [22] in our case) for applicable scripts or rules. The Figure 10 shows the PEP's architecture. A PEP is composed of three modules which are referred to as monitoring, observation and enforcement.

The monitoring module controls the PEP actions and stores all relevant actions/events. It receives abstract policy from the PDP and chooses which action and parameters must be executed to apply the policy. Then, the enforcement module launches this local appropriate action mechanism by applying the selected script. The progress of the operations can be provided to the Observation module. This last module performs periodically, or during a script execution, measurements to evaluate the current state of the PEP. But this is also the module by which an audit is done by sending feedback to the Audit Correlation Engine (ACE).

If we take back the case study presented in section 2, the XACML policy generated in Figure 5 aims at "*allowing subject 26 to read resource 31*". The PEP interfacing with an UNIX-like fileserver registered the

"*setfacl*" action[1]. So it will construct its command by using this action with parameters included into the policy rule. The actions granted by the policy are "read" and "write". They will be transformed into ':rw-' to say that "read" and "write" are allowed but not "execute". The command that the PEP will execute is:

setfacl –m u:26 :rw 31

The "-m" option indicates that the rights are modified, "u" indicates that "26" is a user and ":rw-" are his new rights on 31.

4.3. Audit Correlation Engine

The Audit Correlation Engine goal and architecture is equivalent to the PDP in that it also exhibits its services through the WSIG (Web Service Integration Gateway) and sends policy to the PEP. The ACE receives a request concerning a type of device to audit and/or potentially a resource or a user. As the PDP, it forwards the demand to the concerned PEP related to the request it receives. For that it asks the concerned technical modules to the Facilitator. At the PEP point of view, the policy indicates that this is not a deployment but an audit and for instance, instead of executing a "*setfacl*" command, it executes a "*getfacl*" command in order to get the state of the fileserver concerning a particular resource.

To summarize, the use of a multi-agent system framework gives PDP, PEP and ACE the ability to cooperate and communicate between themselves in order to implement policies and get back their real and current status. It also provides flexibility, openness and heterogeneity because when we decide to add a new PEP, we just have to provide the agent able to concretely apply the policies. This solution provides also interoperability because the services that ACE and PDP offer are exhibited as web service (through the Web Service Integration Gateway, cf. Figure 6) for giving the possibility to the Organisational Layer to communicate with the Technical Layer and also to allow other systems to communicate with this agent-based policy deployment and audit framework. Next section details the links between both layers.

4.4. Links with Organisational model

As explained previously, our approach is based on a twofold development: the generation of access policies from the Organisational Model and their deployment into the different devices by the multi-agents system. Both layers operate in a heterogeneous environment and may consequently be physically or logically distant. Therefore it is necessary to establish communication way disregarding these characteristics. In this context, the most logical and appropriate solution is the use of Web Service. Web Services can meet the needs of interoperability required by SIM. Moreover they are independent and may hence facilitate maintenance without modification of the calls made by clients. The multi-agents system is able to publish all features of its agents through Web Services,. By this way, the link is provided with the Organisational Layer to ensure its monitoring and auditing.

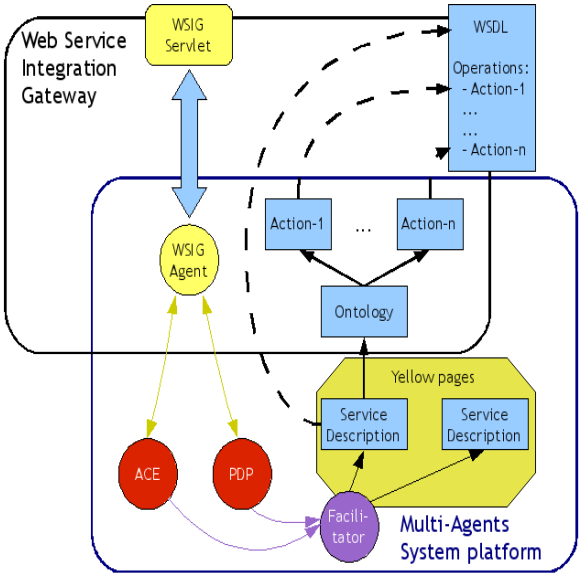

Figure 11: WSIG architecture

As shown in Figure 11, the Web Services Integration Gateway plays the role of web server and so makes the bridge between the multi-agents system and clients (the Organisational Layer). Its main role is to translate all the functionality of agents and Web Services in order to ensure communication with clients. The WSIG interface is composed of two main entities: a web server (the WSIG servlet) and a specialised WSIG agent. When agents register themselves in the yellow pages through the Directory Facilitator, they are also registered in the Service Directory of the WSIG in order to translate them in WSDL format. WSIG agent is able to determine in real-time availability of other agents and all their services to update WSDL files. The Web server gets and forwards the SOAP requests to the WSIG agent, which translates them in ACL messages comprehensible by other agents, notably the PDP. Once agents have completed their work, the result is returned to WSIG, which forwards it to the client. The WSIG model architecture is an add-on of the JADE platform.

5. Conclusion and future work

One means for having *Good IT Governance* is reach by an effective business IT alignment. As a consequence defining policy against business requirements become crucial for business and IT managers. In this paper we

[1] LINUX ACL expands access rights to users and groups. "*setfacl*" and "*getfacl*" are the basic ACL commands.

have presented an architecture developed to applied access rights through the definition of business processes, their transformation into XACML policies and finally their deployment and their audit with a multi-agent system.

The future works will focus on improving three points:

Firstly, our proposed prototype permits to assign rights directly to users. This solution in practices could be difficult to manage if the company encompasses a large number of employees. Solutions exist to face that problem like the usage of role or team to group peoples by function and than affecting rights to it. Our next development will run at integrating that concept in the prototype from the organization to the technical layer.

Secondly, the extension of the XACML policies in order to manage other devices than the fileserver and in order to use a common policy format to deploy and to audit them.

Thirdly, the security of messages exchanged is not taken into account: the messages between agents and Web Service clients are exchanged in plain text format. Malicious users can take advantage could take advantage of this lack of security and may themselves fix rights to various devices to generate their own security policy. As a consequence, we will integrate a two-factor authentication system for Web Service and encryption of messages from agents to ensure the integrity, confidentiality and authenticity of policies.

5. Acknowledgement

SIM “Secure Identity Management” is an R&D project of the CRP Henri Tudor achieved in collaboration with the « University of Luxembourg » funded by the National Research Fund Luxembourg.

7. References

[1] Benjamin Gateau, Christophe Feltus, Jocelyn Aubert, Christophe Incoul, An Agent-based Framework for Identity Management: The Unsuspected Relation with ISO/IEC 15504, IEEE International Conference on Research Challenges in Information Science (IEEE RCIS 2008), Marrakech, Morocco.

[2] Christophe Feltus, Preliminary Literature Review of Policy Engineering Methods - Toward Responsibility Concept, International Conference on Information & Communication Technologies: from Theory to Applications (IEEE ICTTA2008), Damascus, Syria.

[3] David F. Ferraiolo, Ravi Sandhu, Serban Gavrila, D. Richard Kuhn and Ramaswamy Chandramouli, Proposed NIST Standard for Role-Based Access Control, ACM Transactions on Information and System Security, Vol. 4, No. 3, August 2001, Pages 224-274.

[4] Yu, E. S. and Liu, L. 2001. Modelling Trust for System Design Using the i* Strategic Actors Framework. Workshop on Deception, Fraud, and Trust in Agent Societies Held During the Autonomous, Eds. Lecture 35 194.

[5] http://www.egroupware.org

[6] Jean-Pierre Briot and Yves Demazeau, Principes et architectures des systémes multi-agents, Hermés-Lavoisier, 2001.

[7] Nicholas R. Jennings and Michael J. Wooldridge, Applications of intelligent agents, Agent Technology Foundations, Applications, and Markets , Springer-Verlag, 1998.

[8] Simon Godik, Tim Moses, et al, “eXtensible Access Control Markup Language (XACML) Version 1.0”, OASIS Standard, February 18th, 2003.

[9] D. Durham, J. Boyle, R. Cohen, S. Herzog, R. Rajan, A. Sastry, “The COPS (Common Open Policy Service) Protocol”, IETF RFC 2748, january 2000.

[10] R. Enns, “NETCONF Configuration Protocol”, IETF RFC 4741, december 2006.

[11] D. Harrington, R. Presuhn, B. Wijnen, “An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks”, IETF RFC 3411, december 2002.